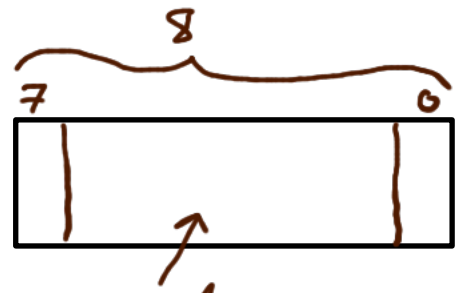
- 8051 has CPU: The CPU contains two types of Registers

① 8-bit registers : A, B, R0, R1, R2, R3, R4, R5, R6, R7, SP, PSW.



Range of values:

min

0000 0000B → 1111 1111B (max)

00H → FFH

0 → 255D

---

② 16-bit registers : PC, DPTR

Range of values

min

0000H → FFFF H (max)

0D → 65535 D

- All previous registers are called the most widely used registers.

---

* A = ACC = Accumulator (8bit)
  - It is must be used in all arithmetic operations (+, −, ×, ÷)
  - In general, it is used in logical operations (AND, OR, ...)
  - Must be used in external data transmision.

---

* B → Auxiliary register
  - It is used an auxiliary register with A in arithmetic operation.
    1. Multiplication (×) → before → hold second operand
       → after → hold high part of result.
    2. Division → before → hold second operand (divisor)
       → after → hold the remainder
       باقي

* R0 ⟶ R7 : general purpose registers

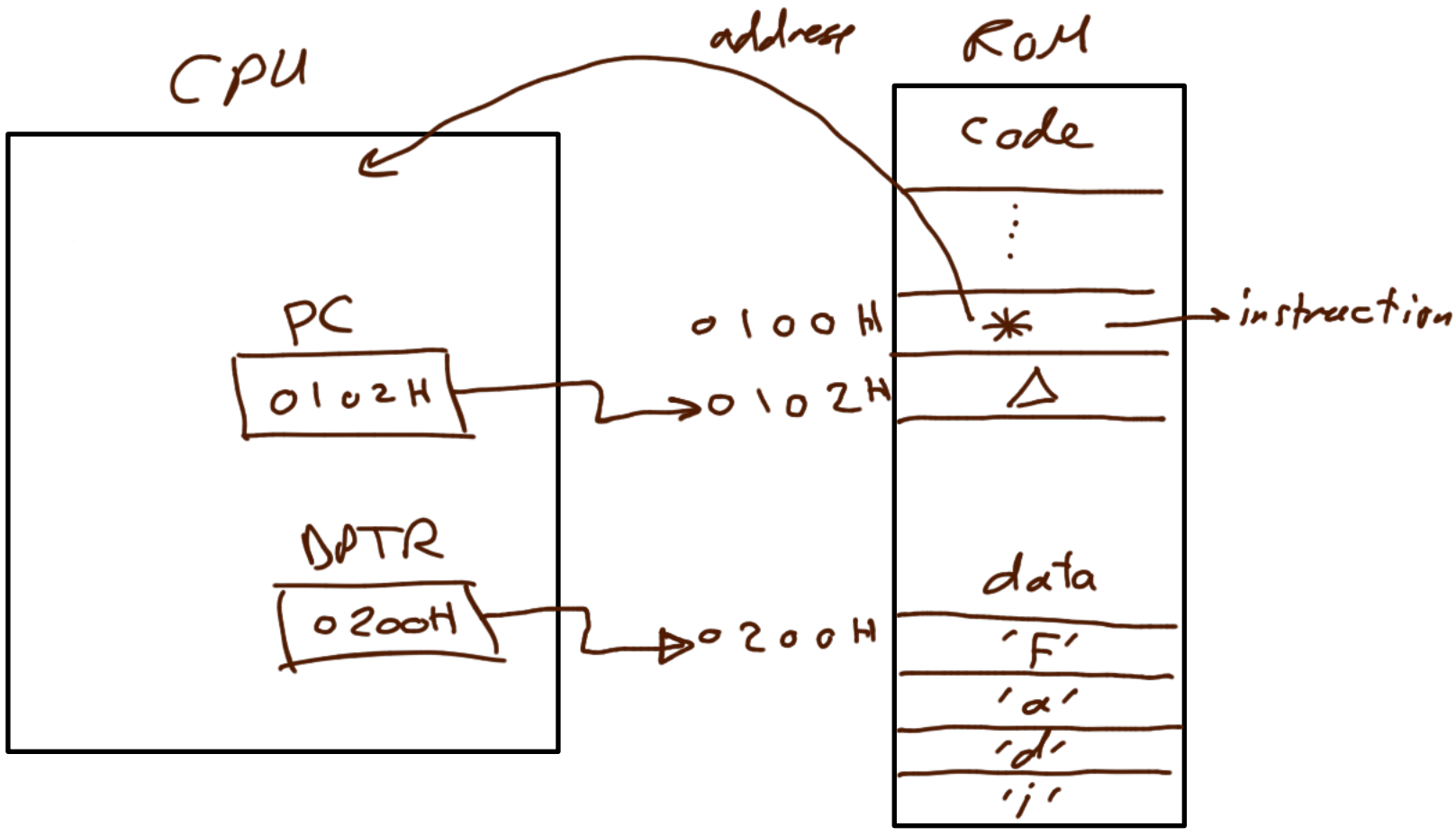
---

* PC : program counter (16 bit)

  - Is used to points to the address of the next instruction that should se fetched from (ROM) memory to be executed in CPU.

* DPTR : Data pointer (16 bit)

  - Is used to point to the address of data that should be fetched from memory to the CPU



- The max size of memory that can be connected with ④

8051: 64kB

- code memory (ROM)
- Data memory (external)

capacity

$$2^{16} = 2^{6} \times 2^{10} = 64\,kB$$

PC → 16 bit

---

address | (ROM) opcode | program

1	0000		ORG 0H	;start at location 0
2	0000	7D25	MOV R5,#25H	;load 25H into R5
3	0002	7F34	MOV R7,#34H	;load 34H into R7
4	0004	7400	MOV A,#0	;load 0 into A
5	0006	2D	ADD A,R5	;add contents of R5 to A ;now A = A + R5
6	0007	2F	ADD A,R7	;add contents of R7 to A ;now A = A + R7
7	0008	2412	ADD A,#12H	;add to A value 12H ;now A = A + 12H
8	000A	80FE HERE:	SJMP HERE	;stay in this loop
9	000C		END	;end of asm source file

PC → 0002

* PSW : program status Register (8 bit)

- Is used to reflects the status of CPU after executing some instructions.
- Each bit in PSW is called flag bit.

7	6	5	4	3	2	1	0
Cy	AC	/	RS1	RS0	OV	-	P

① PSW.7 = C = Cy

- Is carry flag
- Is set (1) when there is carry out from D7.
  ↓ addition

subtraction : carry in D7
borrow

MOV A, # 93 H

ADD A, # 0E2 H

	D_7	D_4	D_3	D_0
	1	0	0	1 1
+	1	1	1	0 0 0
	0	1	1	1 0 1

PSW.7 = C = 1

PSW.6 = AC = 0

② PSW.6 = AC

- Is called auxiliary carry flag (half carry)
- Is set (1) when there is carry from $\underline{D_3}$ to $\underline{D_4}$
- Is used only for BCD addition

---

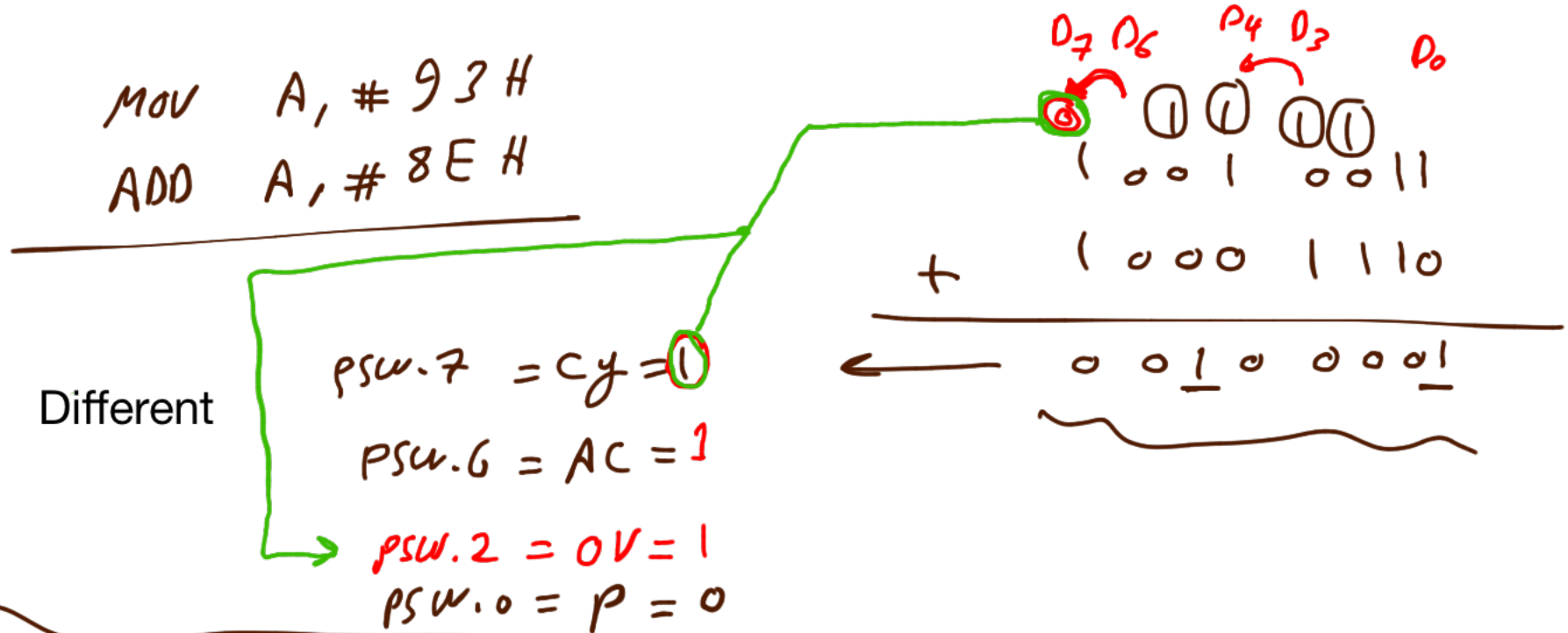
③ PSW.2 = OV

- Is called overflow flag
- Is set (1) to indicate that the result of signed arithmetic operation is error (out of the range). Does not fit in the range.

OV = 1

① when there is carry from D_6 to D_7 and no carry from D_7 to out.

② when there is carry from D_7 to out and no carry from D_6 to D_7.



④ PSW.0 = P

Is parity flag

Is used to detect errors

Is used to reflect number of ones (1's) in A (Accumulator).

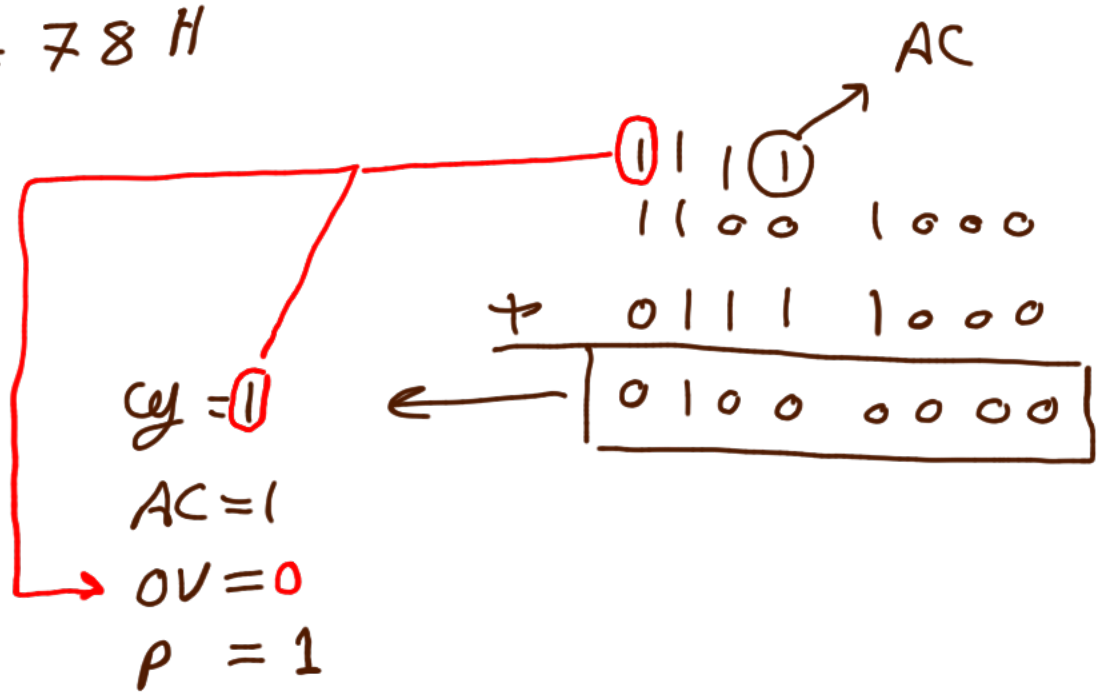
P → =1 if # of 1's in A is odd

=0 " " " = " A is even

Show me the content of PSW (cy, AC, OV, P) (3)

after executing the following

```
MOV  A, #0C8H
AND  A, #78H
```



ADD A, source ; A = A + source

destination

8bit

- affects psw flages (CY, AC, P, OV)

ADD B, #25H X

ADD A, B ✓

ADD B, A X

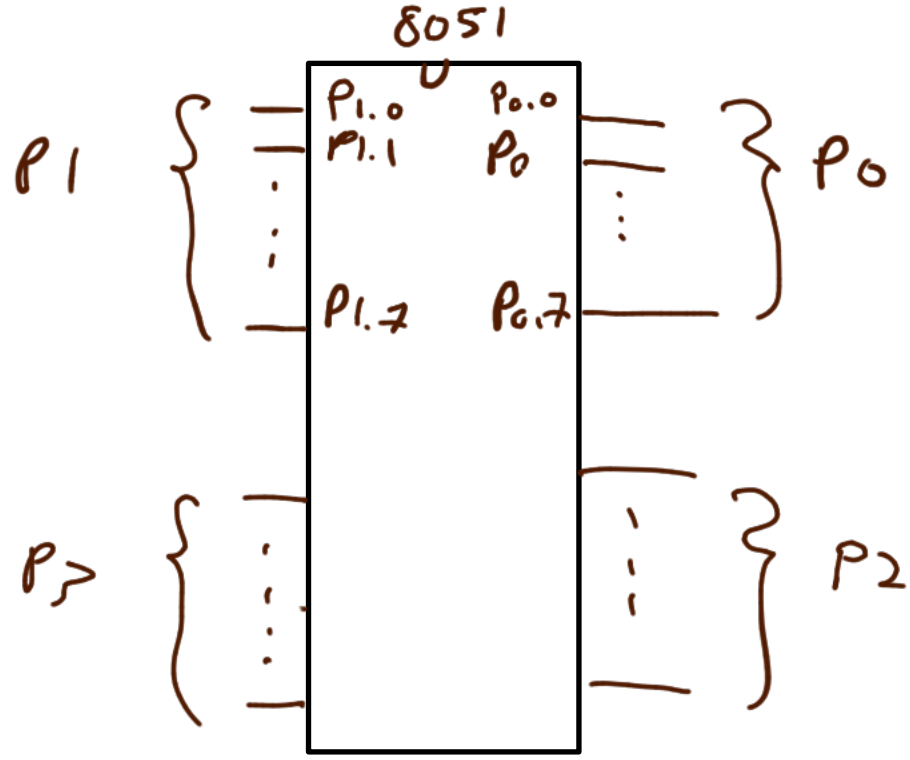
* 8051 features

Table 1-4: Comparison of 8051 Family Members

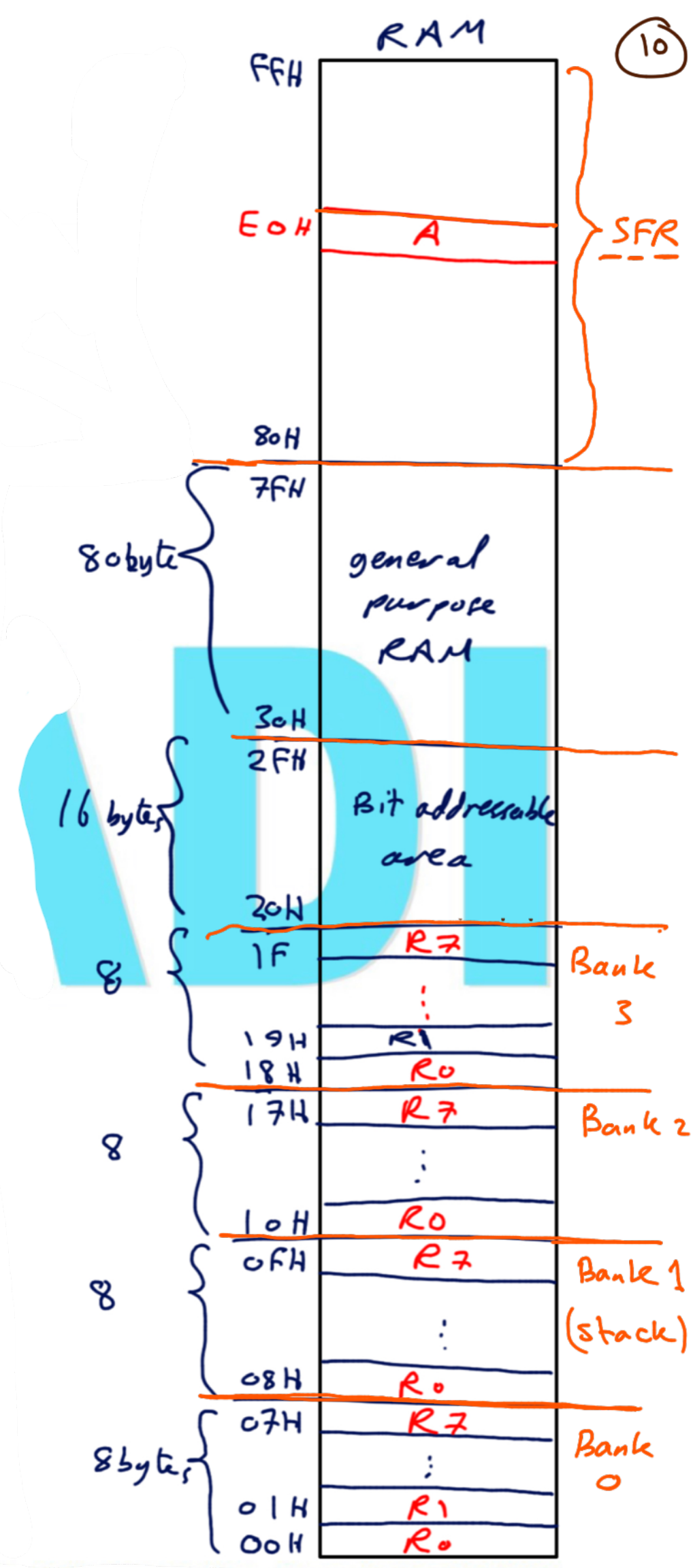
Feature	8051	8052	8031
ROM (on-chip program space in bytes)	4K	8K	0K (0 ROM)
RAM (bytes)	128	256	128
Timers	2	3	2
I/O pins (4 I/O ports)	32	32	32
Serial port	1	1	1
Interrupt sources	6	8	6

each port is 8 pins

8 x 4 = 32 pin



- 8051 is an 8-bit microcontroller


RAM
FFH
E0H
A
SFR
80H
7FH
80byte
general purpose RAM
30H
2FH
16 bytes
Bit addressable area
20H
1FH
8
R7
R1
R0
Bank 3
19H
18H
17H
8
R7
R0
Bank 2
10H
0FH
8
R7
Bank 1 (stack)
08H
R0
07H
8bytes
R7
Bank 0
01H
00H
R1
R0

- Up on reset (power on) all registers (R0 → R7) will be in bank0 (00H ⟶ 07H)

- programmer can move (R0 – R7) from bank0 to bank 1, 2, or 3. using PSW.4 (RS1) and PSW.3(RS0)

RS1 PSW.4	RS0 PSW.3	Bank	
0	0	0	by default
0	1	1	
1	0	2	
1	1	3	

- Move (R0-R7) from bank0 to bank 3.

```
SETB   PSW.4   ;   PSW.4 = RS1 = 1
SETB   PSW.3   ;   PSW.3 = RS0 = 1
```

in bank 3

```
SETB    Bit  ;
```
Bit = 1

```
CLR     Bit  ;
```
Bit = 0

Write a program to move the content of R1 from bank 3 to R7 in to bank 1

```
ORG   0000H
      SETB  PSW.3
      SETB  PSW.4  ; in bank 3
      MOV   A,R1
      SETB  PSW.3
      CLR   PSW.4  ; in bank 1
      MOV   R7, A
      SJMP  $
      END
```